

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

**KHOA VIỄN THÔNG I**

---------🙟🙟🕮🙝🙝---------

****

**BÁO CÁO BÀI TẬP LỚN CUỐI KÌ KỸ THUẬT VI XỬ LÝ**

**Chủ đề: Tìm hiểu STM32 và IC W5500**

**(Giao thức TCP/IP Stack51/STM32/SPI)**

**Giảng Viên**: Nguyễn Trung Hiếu

**Nhóm học – Nhóm con**: 04 – 02

**Thành viên nhóm:**

Trần Duy Anh - B19DCVT018

Ma Đức Cảnh - B19DCVT031

Trịnh Tiến Bình -B19DCVT029

***Hà Nội – 2021***

**LỜI NÓI ĐẦU:**

Trong những năm gần đây, cùng với sự đổi mới phát triển mạnh mẽ của thế giới nói chung và Việt Nam nói riêng thì khoa học và công nghệ cũng đạt được những thành tựu đáng kinh ngạc. Đặc biệt sau cuộc Cách mạng công nghệ 4.0 hàng ngàn các thiết bị điện tử ra đời và ngày càng hoàn thiện đem lại một hiệu suất cao, giá thành hợp lí và giúp ích được rất nhiều cho con người trong các lĩnh vực trong cuộc sống. Do đó các linh kiện điện tử cũng trở nên phổ biến và được nghiên cứu rộng rãi hơn về lợi ích, ưu nhược điểm và cách thức hoạt động của nó. Một trong số đó, chúng ta không thể không kể tới **Module Ethernet W5500 TCP/IP Stack51/STM32 /SPI.** Nó là một mạch giao tiếp sử dụng SPI để kết nối vi điều khiển với Ethernet hiệu quả và ổn định.

Xuất phát từ tính ứng dụng lớn của Module này và định hướng nghiên cứu kĩ hơn về STM32 cùng IC W5500 chúng em xin phép chọn đề tài “Tìm hiểu về STM32 và IC W5500 (giao thức TCP/IP Stack51/STM32 /SPI)” làm đề tài nghiên cứu bài tập lớn cuối kì.

Nhóm em gồm 3 thành viên cùng tham gia nghiên cứu bài tiểu luận:

- Trần Duy Anh (tưởng nhóm): phân công công việc, tìm kiếm nội dung, làm báo cáo.

- Ma Đức Cảnh: tìm kiếm nội dung, làm slide

- Trịnh Tiến Bình: tìm kiếm nội dung

Chúng em xin cảm ơn Thầy Nguyễn Trung Hiếu đã hỗ trợ và tạo điều kiện tốt cho chúng em trong quá trình nghiên cứu và hoàn thành bản báo cáo này. Trong quá trình làm việc sẽ không tránh khỏi những thiếu sót, nhóm em mong có sự góp ý của thầy để hoàn thiện và phát triển hơn.

Nhóm chúng em xin chân thành cảm ơn !

**MỤC LỤC**

[I. TỔNG QUAN VỀ STM32: 4](#_Toc90589381)

[1. Một vài đặc điểm nổi bật của STM32 : 4](#_Toc90589382)

[2. Giới thiệu sơ lược về STM32F103: 5](#_Toc90589383)

[3. Thông số kĩ thuật STM32F103C8T6 : 6](#_Toc90589384)

[II. TỔNG QUAN VỀ IC W5500: 7](#_Toc90589385)

[1. Giới thiệu chung: 7](#_Toc90589386)

[1.1. Các tính năng nổi bật: 8](#_Toc90589387)

[1.2. Ứng dụng IC W5500: 9](#_Toc90589388)

[2. Sơ đồ khối: 10](#_Toc90589389)

[3. Thành phần cấu tạo và chức năng: 11](#_Toc90589390)

[3.1. Pin Assigment: 11](#_Toc90589391)

[3.2. HOST Interface: 12](#_Toc90589392)

[3.2.1. Chế độ hoạt động SPI: 14](#_Toc90589393)

[3.2.2. Khung SPI: 14](#_Toc90589394)

[3.2.3. Chế độ dữ liệu độ dài thay đổi (VGM): 15](#_Toc90589395)

[3.2.4. Chế độ dữ liệu độ dài cố định (FDM): 16](#_Toc90589396)

[3.3. Register and Memory Organization: 17](#_Toc90589397)

[3.3.1. Common Register Block: 18](#_Toc90589398)

[3.3.2. Socket Register Block: 18](#_Toc90589399)

[3.3.3. Memory: 18](#_Toc90589400)

[3.4. Register Descriptions: 19](#_Toc90589401)

[3.4.1. Common Register: 19](#_Toc90589402)

[3.4.2. Socket Registers: 22](#_Toc90589403)

[III. MODULE ETHERNET W5500–GIAO THỨC TCP/IP Stack51/STM32/SPI: 28](#_Toc90589404)

[1. Tìm hiểu chung về các giao thức: 28](#_Toc90589405)

[1.1. TCP/IP: 28](#_Toc90589406)

[1.2. STM32/SPI: 32](#_Toc90589407)

[2. Công nghệ & Tính năng sản phẩm thực tế: 37](#_Toc90589408)

[3. Thông số kỹ thuật sản phẩm: 37](#_Toc90589409)

[IV. TÀI LIỆU THAM KHẢO: 38](#_Toc90589410)

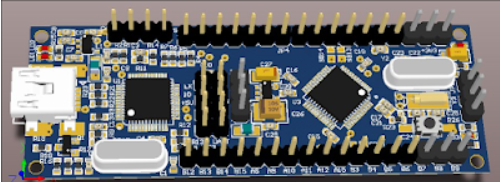
# **I. TỔNG QUAN VỀ STM32:**

## **1. Một vài đặc điểm nổi bật của STM32 :**

* Vi điều khiển STM32 dựa trên nền lõi ARM Cortex-M3 thế hệ mới do hãng ARM thiết kế sử dụng lõi ARM Cortex – M3 là sự cải tiến của ARM7.
* Một vài đặc điểm nổi bật của STM32 :
* ***Sự tinh vi*** : STM32 hỗ trợ thêm 7 kênh DMA. Mỗi kênh được dùng để truyền dữ liệu đến các thanh ghi ngoại vi hoặc từ các thanh ghi ngoại vi đi với kích thước từ dữ liệu có thể truyền là 8/16 hoặc 32-bit. Các đơn vị DMA dễ dùng, linh hoạt và tự động điều khiển các luồng dữ liệu bên trong vi điều khiển. STM32 là một vi điều khiển tiêu thụ năng lượng thấp và đạt hiệu suất cao. Khả năng nhanh đi vào và thoát khỏi các chế độ tiết kiệm năng lượng làm giảm nhiều sự tiêu thụ năng lượng tổng thể.
* ***Sự an toàn*** :STM32 cung cấp một số tính năng phần cứng hỗ trợ các ứng dụng một cách tốt nhất, bao gồm: một bộ phát hiện điện áp thấp, một hệ thống bảo vệ xung clock và hai bộ watchdogs.
* ***Tính bảo mật*** :Bộ nhớ Flash của STM32 có thể được khóa để chống truy cập đọc Flash thông qua cổng debug. Khi tính năng bảo vệ đọc được kích hoạt, bộ nhớ Flash cũng được bảo vệ chống ghi để ngăn chặn mã không tin cậy được chèn vào bảng vector ngắt. STM32 cũng có một đồng hồ thời gian thực và một khu vực nhỏ dữ liệu trên SRAM được nuôi nhờ nguồn pin. Khu vực này có một đầu vào chống giả mạo, có thể kích hoạt một sự kiện ngắt khi có sự thay đổi trạng thái ở đầu vào này. Ngoài ra một sự kiện chống giả mạo sẽ tự động xóa dữ liệu được lưu trữ trên  
  SRAM được nuôi bằng nguồn pin.
* ***Phát triển phần mềm*** : hệ thống CoreSight trên STM32 cung cấp một data watchpoint và một công cụ theo dõi (instrumentation trace). Công cụ này có thể gửi thông tin về ứng dụng được lựa chọn đến công cụ gỡ lỗi. Điều này có thể cung cấp thêm các thông tin gỡ lỗi và cũng có thể được sử dụng trong quá trình thử nghiệm phần mềm.
* ***Dòng Performance và Access của STM32***: họ STM32 có hai nhanh riêng biệt: dòng Performance và dòng Access. Dòng Performance tập hợp đầy đủ các thiết bị ngoại vi và chạy ở xung nhịp tối đa 72MHz. Dòng Access có các thiết bị ngoại vi ít hơn và chạy ở 32MHz. Cách bố trí chân pin và cách đóng gói chip là như nhau cho phép các phiên bản khác nhau của STM32 được hoán vị mà không cần phải sửa đổi lại footprint (mô hình chân của chip trong công cụ layout bo mạch) trên PCB.

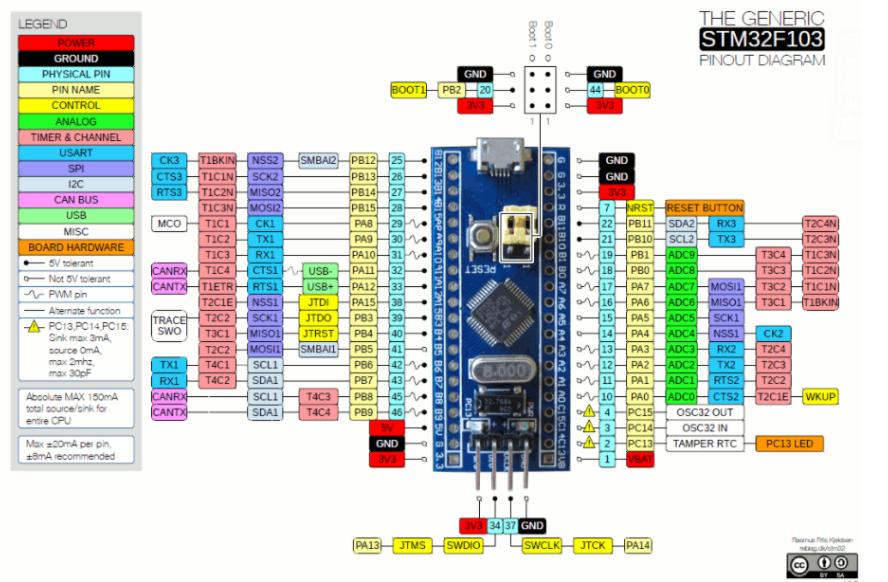
## **2. Giới thiệu sơ lược về STM32F103:**

* STM32 là một trong những dòng chip phổ biến của ST với nhiều họ thông dụng như F0, F1, F2, F3, F4… STM32F103 thuộc họ F1 với lõi là ARM COTEX M3. STM32F103 là vi điều khiển 32 bit, tốc độ tối đa là 72Mhz. Giá thành cũng khá rẻ so với các loại vi điều khiển có chức năng tương tự.
* Một số ứng dụng chính: dùng cho driver để điều khiển ứng dụng, điều khiển ứng dụng thông thường, thiết bị cầm tay và thuốc, máy tính và thiết bị ngoại vi chơi game, GPS cơ bản, các ứng dụng trong công nghiệp, thiết bị lập trình PLC, biến tần, máy in, máy quét, hệ thống cảnh báo, thiết bị liên lạc nội bộ…
* Phần mềm lập trình: IAR Embedded Workbench, KeilC…
* Thư viện lập trình: có nhiều loại thư viện lập trình cho STM32 như: STM32snippets, STM32Cube LL, STM32Cube HAL, Standard Peripheral Libraries, Mbed core…
* Sơ lược về Board trên:



* 1 cổng Mini USB dùng để cấp nguồn, nạp cũng như debug.
* 2 MCU bao gồm 1 MCU nạp và 1 MCU dùng để lập trình.
* Có chân Output riêng cho mạch nạp trên MCU1.
* Có chân Output đầy đủ cho các chân MCU2.
* Chân cấp nguồn ngoài riêng cho MCU2 nếu không sử dụng nguồn từ USB.
* Thạch anh 32,768khz dùng cho RTC và Backup.
* Chân nạp dùng cho chế độ nạp boot loader.
* Nút Reset ngoài và 1 led hiển thị trên chân PB9, 1 led báo nguồn cho MCU2.

## **3. Thông số kĩ thuật STM32F103C8T6** **:**



* Vi điều khiển: STM32F103C8T6.
* Điện áp cấp 5VDC qua cổng Micro USB sẽ được chuyển đổi thành 3.3VDC qua IC nguồn và cấp cho Vi điều khiển chính.
* Tích hợp sẵn thạch anh 8Mhz.
* Tích hợp sẵn thạnh anh 32Khz cho các ứng dụng RTC.
* Ra chân đầy đủ tất cả các GPIO và giao tiếp: CAN, I2C, SPI, UART, USB,…
* Tích hợp Led trạng thái nguồn, Led PC13, Nút Reset.
* Kích thước: 53.34 x 15.24mm
* Sử dụng với các mạch nạp:
* ST – Link Mini
* J -Link
* USB TO COM
* Kết nối chân khi nạp bằng ST-Link Mini
* Nạp theo chuẩn SWD
* TCK – SWCLK
* TMS – SWDIO
* GND – GND
* 3.3V – 3.3V

# **II. TỔNG QUAN VỀ IC W5500:**

## **1. Giới thiệu chung:**

- Chip W5500 là bộ điều khiển Ethernet nhúng TCP/IP có dây cứng giúp kết nối Internet dễ dàng hơn đối với các hệ thống nhúng.

- W5500 cho phép người dùng có khả năng kết nối Internet trong các ứng dụng của họ chỉ bằng cách sử dụng một chip duy nhất có ngăn xếp TCP/IP, 10/100 Ethernet MAC và PHY được nhúng.

- Hardwired TCP/IP hỗ trợ các giao thức TCP, UDP, IPv4, ICMP, ARP, IGMP và PPPoE. W5500 nhúng bộ đệm nhớ trong 32Kbyte để xử lý gói Ethernet.

- Khi dùng W5500 có thể triển khai ứng dụng Ethernet chỉ bằng cách thêm chương trình ổ cắm đơn giản, có thể sử dụng đồng thời 8 ổ cắm phần cứng độc lập.

- SPI được cung cấp để dễ dàng tích hợp với MCU bên ngoài. SPI của W5500 hỗ trợ tốc độ 80MHz và giao thức SPI hiệu quả mới cho giao tiếp mạng tốc độ cao.

- W5500 cung cấp thế độ WOL (Wake on LAN) và tắt nguồn.



*IC W5500*

### **1.1. Các tính năng nổi bật:**

**-** Hỗ trợ các giao thức TCP / IP có dây cứng: TCP, UDP, ICMP, IPv4, ARP, IGMP, PPPoE.

- Hỗ trợ đồng thời 8 ổ cắm độc lập.

- Hỗ trợ chế độ tắt nguồn.

- Hỗ trợ Wake on LAN qua UDP

- Hỗ trợ giao diện ngoại vi nối tiếp tốc độ cao (SPI MODE 0,3)

- Bộ nhớ 32Kbyte bên trong cho bộ đệm TX/RX

- 10BaseT / 100BaseTX Ethernet PHY nhúng

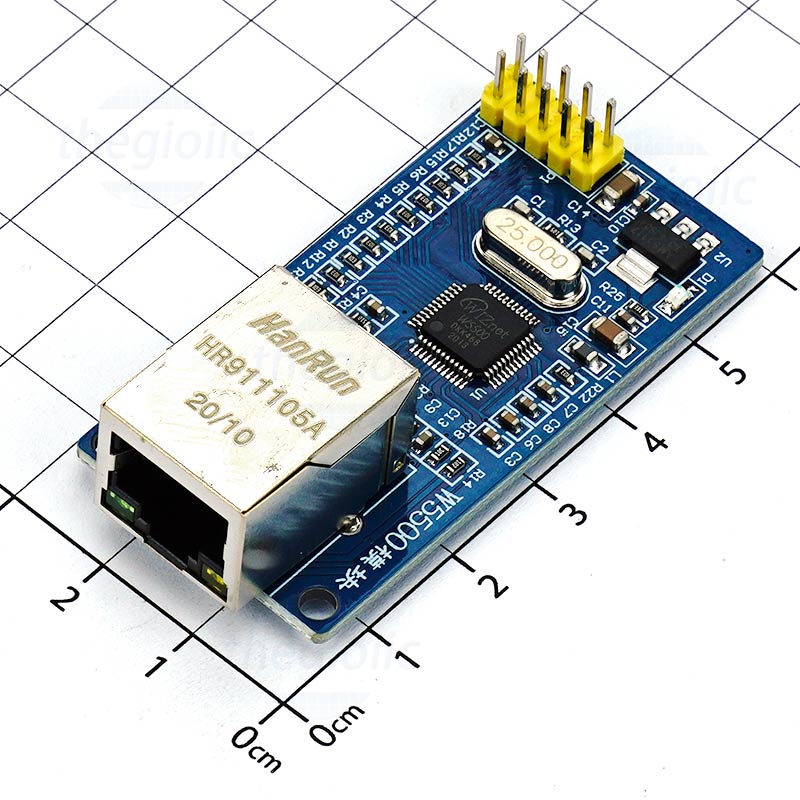
- Hỗ trợ thương lượng tự động (Full and half duplex, 10 and 100-based)

- Không hỗ trợ phân mảnh IP

- Hoạt động 3.3V với dung sai tín hiệu I/O 5V

- Đầu ra LED (Full / Half duplex, Link, Speed, Active)

- Gói 48 Pin LQFP (7x7 mm).



*W5500 chuyển giao tiếp Ethernet SPI*

### **1.2. Ứng dụng IC W5500:**

- Thiết bị mạng gia đình: Hộp giải mã tín hiệu, PVR, bộ điều hợp phương tiện kĩ thuật số.

- Nối tiếp Ethernet: Điều khiển truy cập, hiển thị LED, Wireless AP…

- Song song với Ethernet: Máy POS/ Máy in Mini, máy photocopy.

- USB – to – Ethernet : thiết bị lưu trữ, máy in mạng.

- GPIO – to – Ethernet: Cảm biến mạng gia đình.

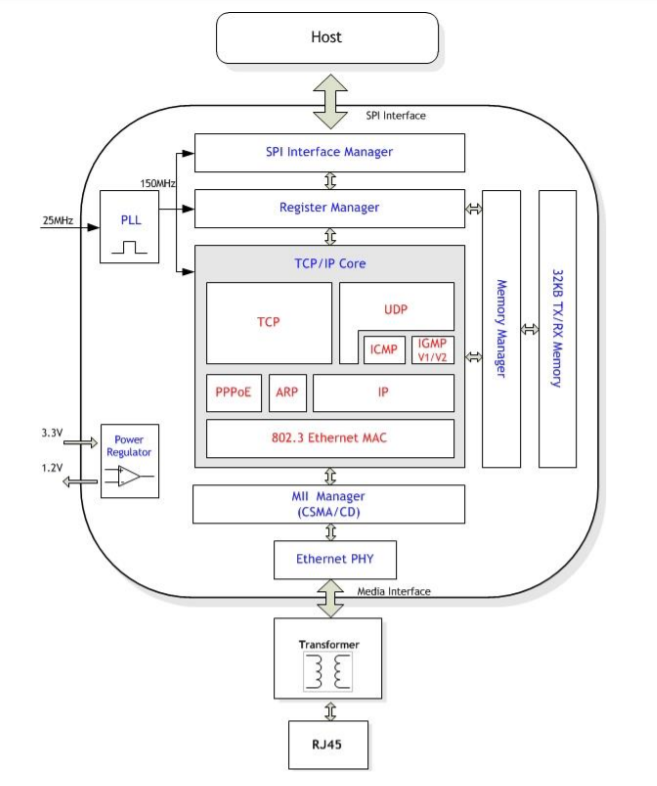
- Hệ thống an ninh: DVR, Camera mạng, Kiosk.

- Tự động hóa nhà máy và tòa nhà.

- Thiết bị theo dõi y tế.

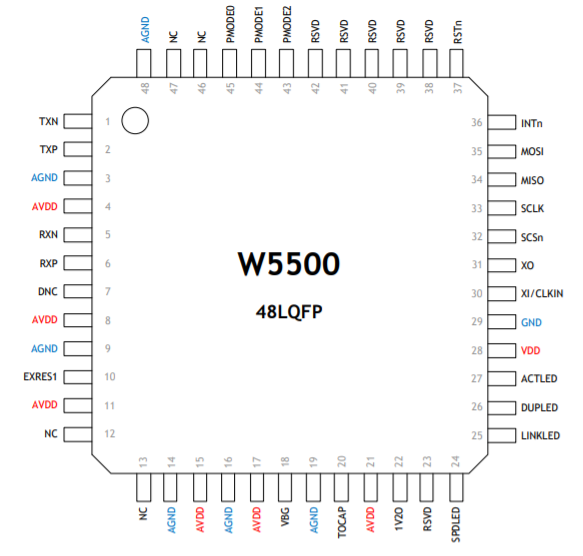
- Máy chủ nhúng.

## **2. Sơ đồ khối:**

****

## **3. Thành phần cấu tạo và chức năng:**

### **3.1. Pin Assigment:**

****

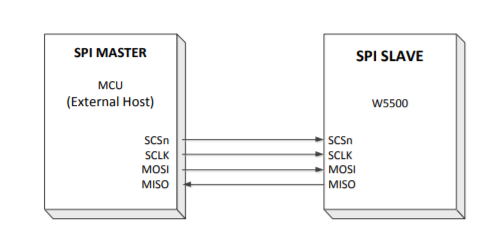
* Pin 1: Cặp tín hiệu TXP / TXN
* Pin 2: Dữ liệu khác biệt được truyền đến phương tiện truyền thông trên TXP / TXN.
* Pin 3: Analog ground
* Pin 4: Analog 3.3V power
* Pin 5: Cặp tín hiệu RXP/ RXN
* Pin 6: Dữ liệu khác biệt từ phương tiện được nhận trên cặp tín hiệu RXP / RXN.
* Pin 7: Không kết nối mã pin
* Pin 8: Analog 3.3 power
* Pin 9: Analog ground
* Pin 10: Điện trở tham chiếu bên ngoài
* Pin 11: Analog 3.3V power
* Pin 12&13: NC
* Pin 14: Analog ground
* Pin 15: Analog 3.3V power
* Pin 16: Analog ground
* Pin 17: Analog 3.3V power
* Pin 18: Chân này sẽ được đo là 1,2V ở 25 ℃.Nó phải được thả nổi.
* Pin 19: Analog ground
* Pin 20: Tụ điện tham chiếu bên ngoài
* Pin 21: Analog 3.3V power
* Pin 22: Điện áp đầu ra bộ điều chỉnh 1.2V
* Pin 23: Được gắn với GND.
* Pin 24: Speed LED (Low: 100Mbps, High: 10Mbps)
* Pin 25: Link LED
* Pin 26: Duplex LED
* Pin 27: Active LED
* Pin 28: Digital 3.3V Power
* Pin 29: Digital Ground
* Pin 30: Crystal input / External Clock input
* Pin 31: Crystal output
* Pin 32: Chip Select for SPI bus (Low: selected, High: deselected)
* Pin 33: SPI clock input
* Pin 34: SPI master input slave(W5500) output
* Pin 35: SPI master output slave(W5500) input
* Pin 36: Interrupt output (Low: Interrupt asserted from W5500; High: No interrupt)
* Pin 37: Reset (Active Low)
* Pin 38 – 42: NC
* Pin 43- 44 - 45 : Các chân PHY chọn chế độ hoạt động
* Pin 46 & 47: NC
* Pin 48: Analog ground

### **3.2. HOST Interface:**

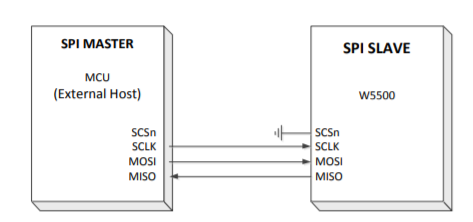
**-** W5500 cung cấp giao diện Bus SPI với 4 tín hiệu (SCSn, SCLK, MOSI, MISO) cho giao diện HOST bên ngoài và hoạt động như một SPI Slave.

- W5500 được kết nối với MCU (STM32) như hình 3.2.1 và hình 3.2.2 theo chế độ hoạt động của nó.

- Trong hình 3.2.1, Bus SPI có thể được chia sẻ với các thiết bị khác. Vỉ Bus SPI dành riêng cho W5500, không chia sẻ Bus SPI với các thiết bị SPI khác nó được thể hiện ở hình 3.2.2

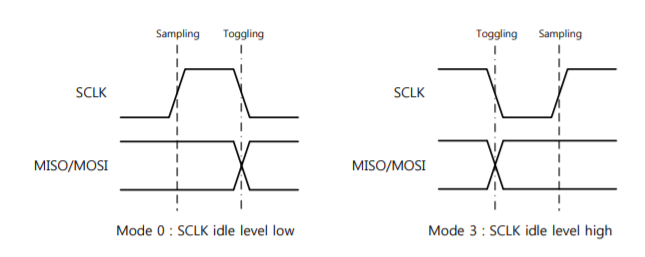


*Hình 3.2.1 chế độ dữ liệu độ dài thay đổi (SCSn do máy chủ điều khiển)*

**

*Hình 3.2.2 chế độ dữ liệu độ dài được cố định (SCSn được kết nối với đất)*

- W5500 hỗ trợ Chế độ SPI 0 và Chế độ 3. Cả tín hiệu MOSI và MISO đều sử dụng trình tự chuyển từ Bit quan trọng nhất (MSB) sang Bit quan trọng nhất (LSB) khi tín hiệu MOSI truyền đi và tín hiệu MISO nhận. Tín hiệu MOSI & MISO luôn truyền hoặc nhận theo trình tự từ Bit quan trọng nhất (MSB) đến Bit quan trọng nhất(LSB).



*Hình 3.3.3 SPI mode 0 & 3*

#### **3.2.1. Chế độ hoạt động SPI:**

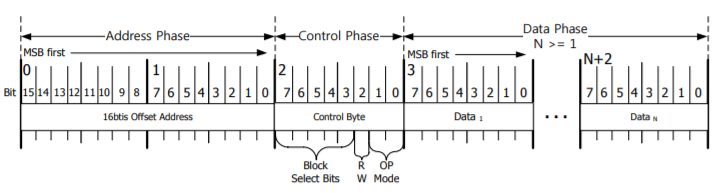
- W5500 được điều khiển bởi khung SPI giao tiếp với máy chủ bên ngoài. Khung W5500 SPI gồm ba pha: giai đoạn địa chỉ, giai đoạn dữ liệu, giai đoạn kiểm soát.

- Nếu chế độ hoạt động SPI là chế độ dữ liệu độ dài thay đổi (VDM) thì tín hiệu Bus SPI SCSn được điều khiển bởi máy chủ bên ngoài với các bước khung SPI.

#### **3.2.2. Khung SPI:**

- W5500 SPI Frame bao gồm địa chỉ bù 16 bits trong pha địa chỉ, 8bits trong pha điều khiển.

- Giai đoạn và N byte Giai đoạn dữ liệu được thể hiện trong hình 3.2.2.1. Giai đoạn điều khiển 8bits được cấu hình lại với các bit Chọn khối (BSB [4: 0]), Đọc / Ghi . Chế độ truy cập bit (RWB) và Chế độ hoạt động SPI (OM [1: 0]).



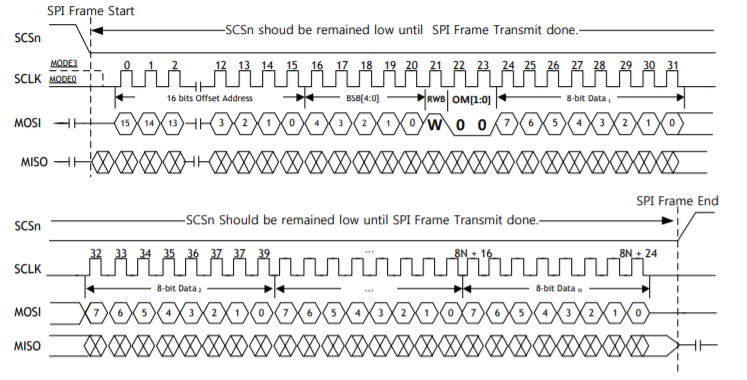
*Hình 3.2.2.1 định dạng khung SPI*

* ***Giai đoạn địa chỉ***: Giai đoạn Địa chỉ này chỉ định Địa chỉ bù 16 bit cho các thanh ghi W5500 và Khối đệm TX / RX.Giá trị Địa chỉ bù 16 bit được chuyển tuần tự từ MSB sang LSB. Khung SPI với pha dữ liệu 2/4 / N byte hỗ trợ Đọc / Ghi dữ liệu tuần tự trong mà địa chỉ Offset sẽ tự động tăng 1 mỗi dữ liệu 1 byte.
* ***Giai đoạn kiểm soát***: Giai đoạn điều khiển chỉ định Khối mà Địa chỉ bù đắp (đặt theo Địa chỉPha) thuộc về Chế độ Truy cập Đọc / Ghi và Chế độ Hoạt động SPI.
* ***Giai đoạn dữ liệu***: Giai đoạn dữ liệu được đặt theo hai loại độ dài, một loại là độ dài N-Byte (chế độ VDM) và một loại là độ dài loại là 1/2/4 Byte (chế độ FDM).Tại thời điểm này, 1 byte dữ liệu được chuyển qua MOSI hoặc MISO tín hiệu từ MSB đến LSB một cách tuần tự.

#### **3.2.3. Chế độ dữ liệu độ dài thay đổi (VGM):**

- Trong chế độ VDM, Độ dài Giai đoạn Dữ liệu Khung SPI được xác định bởi Điều khiển SCSn của Máy chủ bên ngoài. Điều đó có nghĩa là Độ dài giai đoạn dữ liệu có thể có giá trị ngẫu nhiên (Bất kỳ độ dài từ 1 Byte đến N Byte) theo Điều khiển SCSn.OM [1: 0] của Giai đoạn Điều khiển phải là giá trị '00' trong chế độ VDM.

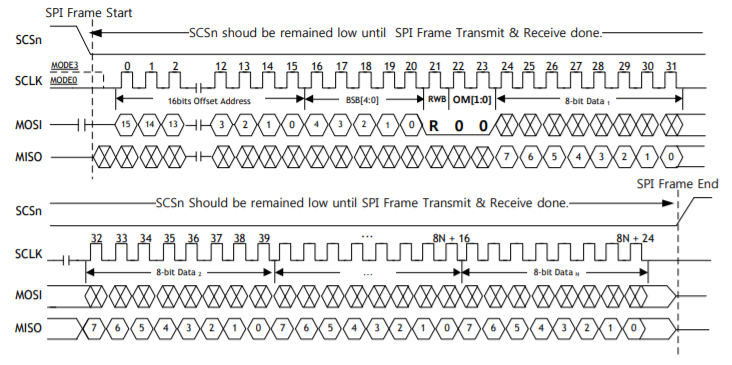
* Truy cập ghi trong VDM:



*Hình 3.2.3.1 Ghi Khung SPI ở chế độ VDM*

Hình 3.2.3.1 cho thấy Khung SPI khi máy chủ bên ngoài truy cập W5500 choviết.Ở chế độ VDM, tín hiệu RWB là '1' (Ghi), OM [1: 0] là '00' trong Giai đoạn Điều khiển khung hình SPI. Tại thời điểm này, máy chủ bên ngoài xác nhận tín hiệu SCSn (Cao đến Thấp) trước đótruyền Khung SPI. Sau đó Host truyền tất cả các bit của SPI Frame tới W5500 thông qua tín hiệu MOSI. Tất cả các bit được đồng bộ hóa với cạnh xuống của SCLK. Sau khi kết thúc quá trình truyền khung SPI, máy chủ lưu dữ liệu SCSn.

* Truy cập đọc trong VDM:



*Hình 3.2.3.2 đọc khung SPI ở chế độ VDM*

Hình 3.2.3.2 cho thấy Khung SPI khi máy chủ bên ngoài truy cập W5500 để đọc ở chế độ VDM, tín hiệu RWB là '0' (Ghi), OM [1: 0] là '00' trong Giai đoạnđiều khiển khung hình SPI. Tại thời điểm này, máy chủ bên ngoài xác nhận tín hiệu SCSn (Cao đến Thấp) trước đótruyền Khung SPI.Sau đó Máy chủ truyền Địa chỉ và Giai đoạn Điều khiển tất cả các bit đến W5500 thông qua MOSIdấu hiệu. Tất cả các bit được đồng bộ hóa với cạnh rơi của SCLK.Sau đó Máy chủ nhận tất cả các bit của Giai đoạn dữ liệu với việc đồng bộ hóa cạnh lên củaLấy mẫu SCLK thông qua tín hiệu MISO.Sau khi kết thúc nhận Giai đoạn dữ liệu, Máy chủ hủy xác nhận tín hiệu SCSn (Thấp đếnCao).

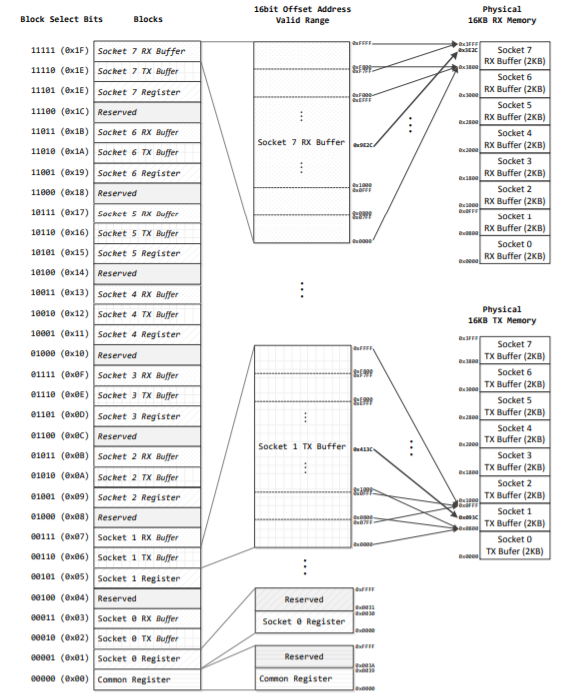
#### **3.2.4. Chế độ dữ liệu độ dài cố định (FDM):**

**-** Chế độ FDM có thể được sử dụng khi Máy chủ Bên ngoài không thể điều khiển tín hiệu SCSn.Tín hiệu SCSn phải được gắn với Thấp (Luôn kết nối với GND) và không thể để chia sẻ Bus SPI với các Thiết bị SPI khác.Trong chế độ VDM, độ dài Giai đoạn Dữ liệu được điều khiển bởi điều khiển SCSn.Nhưng ở chế độ FDM, độ dài Pha dữ liệu được điều khiển bởi giá trị OM [1: 0] ('01' / '10' / '11')là Bit Chế độ Hoạt động SPI của Giai đoạn Điều khiển.Vì Khung SPI của chế độ FDM giống với Khung SPI của chế độ VDM (1Byte, 2 byte,4 byte SPI Frame) ngoại trừ điều khiển tín hiệu SCSn và cài đặt OM [1: 0].

### **3.3. Register and Memory Organization:**

- W5500 có một khối thanh ghi chung: tám khối thanh ghi ổ cắm TX / RX, khối đệm được phân bố cho mỗi Socket.

- Hình 3.3.1 cho thấy khối được chọn bởi BSB [4: 0] và địa chỉ bù sẵn cóphạm vi của Khối đệm Socket TX / RX. Mỗi Khối đệm TX của Socket tồn tại trong một 16KBBộ nhớ TX vật lý và được cấp phát ban đầu với 2KB.Ngoài ra, Khối đệm RX của mỗi Socket tồn tại trong một Bộ nhớ RX 16KB về mặt vật lý và làđược phân bổ ban đầu với 2KB.Bất kể kích thước được phân bổ của mỗi Bộ đệm Socket TX / RX, nó có thể được truy cậptrong phạm vi địa chỉ bù 16 bit (Từ 0x0000 đến 0xFFFF).



*Hình 3.3.1 Register and Memory Organization*

#### **3.3.1. Common Register Block:**

- Khối Đăng ký Chung cấu hình thông tin chung của W5500 như IP và Địa chỉ MAC. Khối này có thể được chọn bởi giá trị BSB [4: 0] của khung SPI.

#### **3.3.2. Socket Register Block:**

- W5500 hỗ trợ 8 socket cho kênh giao tiếp. Mỗi Socket được điều khiển bởi Socket n Khối thanh ghi (khi 0≤n≤7). Giá trị n của thanh ghi Socket n có thể được chọnbởi BSB [4: 0] của khung SPI.

#### **3.3.3. Memory:**

- W5500 có một khối nhớ TX 16KB cho khối đệm Socket n TX và một bộ nhớ 16KB RX bộ nhớ cho khối đệm Socket n RX.

- Bộ nhớ 16KB TX ban đầu được phân bổ ở kích thước 2KB cho mỗi Khối đệm TX Socket (2KBX 8 = 16KB). Kích thước 2KB được phân bổ ban đầu của Bộ đệm Socket n TX có thể được phân bổ lại bằngbằng cách sử dụng 'Thanh ghi kích thước bộ đệm Socket n TX (Sn\_TXBUF\_SIZE)'.

- Khi tất cả các thanh ghi Sn\_TXBUF\_SIZE đã được định cấu hình, Bộ đệm Socket TX được cấp phát với kích thước đã định cấu hình của Bộ nhớ TX 16KB và được gán tuần tự từ Socket0 đến Socket 7. Địa chỉ bộ nhớ vật lý của nó được tự động xác định trong 16KB TX bộ nhớ. Do đó, tổng số Sn\_TXBUF\_SIZE không được vượt quá 16 trong trường hợplỗi trong quá trình truyền dữ liệu.

- Khối đệm Socket n TX được phân bổ trong bộ nhớ TX 16KB là bộ đệm để lưu dữ liệuđược truyền bởi máy chủ. Địa chỉ bù 16bits của Khối đệm Socket n TX có không gian địa chỉ 64KB dao động từ 0x0000 đến 0xFFFF và nó được định cấu hình với tham chiếuđến 'Thanh ghi con trỏ ghi Socket n TX (Sn\_TX\_WR)' & 'Con trỏ đọc Socket n TXđăng ký (Sn\_RX\_RD) '. Tuy nhiên, Địa chỉ bù đắp 16bits tự động chuyển đổi thànhđịa chỉ vật lý có thể truy cập được trong bộ nhớ TX 16KB.

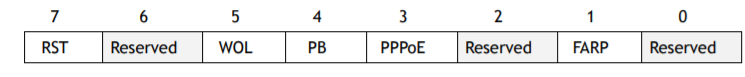
- Khối đệm Socket n RX được phân bổ trong bộ nhớ 16KB RX là bộ đệm để lưunhận dữ liệu qua Ethernet. Địa chỉ bù 16bits của bộ đệm Socket n RXKhối có không gian địa chỉ 64KB nằm trong khoảng từ 0x0000 đến 0xFFFF và nó được cấu hình vớitham chiếu đến 'Thanh ghi con trỏ RD Socket n RX (Sn\_RX\_RD)' & 'Con trỏ ghi Socket n RXĐăng ký (Sn\_RX\_WR) '. Tuy nhiên, Địa chỉ bù đắp 16bits tự động chuyển đổi thànhđịa chỉ vật lý có thể truy cập trong bộ nhớ 16KB RX.

### **3.4. Register Descriptions:**

#### **3.4.1. Common Register:**

***MR (Đăng ký chế độ) [R / W] [0x0000] [0x00]***

MR được sử dụng để đặt lại S / W, chế độ khối ping và chế độ PPPoE.



***GAR (Đăng ký địa chỉ IP cổng) [R / W] [0x0001 - 0x0004] [0x00]***

GAR định cấu hình địa chỉ cổng mặc định.

***SUBR (Đăng ký mặt nạ mạng con) [R / W] [0x0005 - 0x0008] [0x00]***

SUBR định cấu hình địa chỉ mặt nạ mạng con.

***SHAR (Đăng ký địa chỉ phần cứng nguồn) [R / W] [0x0009 - 0x000E] [0x00]***

SHAR định cấu hình địa chỉ phần cứng nguồn.

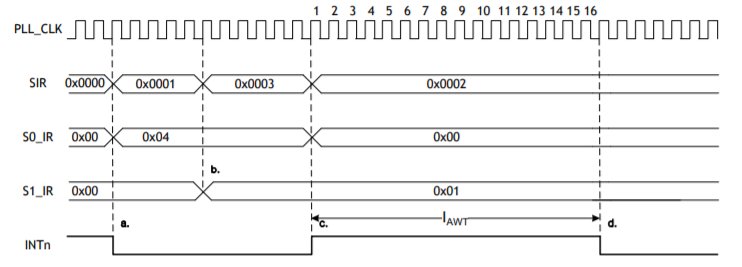
***SIPR (Đăng ký địa chỉ IP nguồn) [R / W] [0x000F - 0x0012] [0x00]***

SIPR cấu hình địa chỉ IP nguồn.

***INTLEVEL (Thanh ghi hẹn giờ mức thấp ngắt) [R / W] [0x0013 - 0x0014] [0x0000]***

INTLEVEL định cấu hình Thời gian chờ thông báo ngắt (I AWT ). Khi gián đoạn tiếp theoxảy ra, mã PIN ngắt (INTn) sẽ xác nhận ở mức thấp sau thời gian INTLEVEL.





*INTLEVEL Timing*

***IR (Thanh ghi ngắt) [R / W] [0x0015] [0x00]***

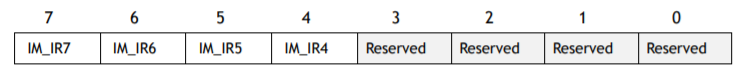
IR cho biết trạng thái ngắt. Mỗi bit IR có thể bị xóa khi máy chủ ghiGiá trị '1' cho mỗi bit. Nếu IR không bằng '0x00', mã PIN INTn được khẳng định ở mức thấp cho đến khi

'0x00'.



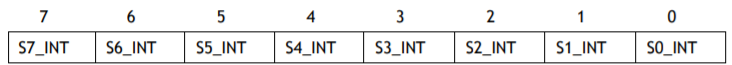
***IMR (Thanh ghi mặt nạ ngắt) [R / W] [0x0016] [0x00]***

IMR được sử dụng để che dấu ngắt. Mỗi bit IMR tương ứng với mỗi bit IR. Khi nàomột bit của IMR là '1' và bit IR tương ứng là '1', một ngắt sẽ được đưa ra. Trongnói cách khác, nếu một bit IMR là '0', thì ngắt sẽ không được đưa ra ngay cả khibit IR tương ứng là '1'.



***SIR (Thanh ghi ngắt ổ cắm) [R / W] [0x0017] [0x00]***

SIR cho biết trạng thái ngắt của Socket. Mỗi bit của SIR vẫn là '1' cho đến khi Sn\_IR làđược xóa bởi máy chủ. Nếu Sn\_IR không bằng '0x00', bit thứ n của SIR là '1' và INTn.



***SIMR (Thanh ghi mặt nạ ngắt ổ cắm) [R / W] [0x0018] [0x00]***

Mỗi bit của SIMR tương ứng với mỗi bit của SIR. Khi một bit của SIMR là '1' vàbit tương ứng của SIR là '1', ngắt sẽ được đưa ra. Nói cách khác, nếu một chút SIMRlà '0', một ngắt sẽ không được cấp ngay cả khi bit tương ứng của SIR là '1'.



***RTR (Thử lại thanh ghi giá trị thời gian) [R / W] [0x0019 - 0x001A] [0x07D0]***

RTR định cấu hình khoảng thời gian chờ truyền lại. Đơn vị của khoảng thời gian chờ là 100us và mặc định của RTR là '0x07D0' hoặc '2000'. Và khoảng thời gian chờ mặc định là 200ms (100us X 2000).Trong thời gian được cấu hình bởi RTR, W5500 sẽ đợiphản hồi của đồng đẳng đối với góiđược truyền bởi Sn\_CR (CONNECT, DISCON, CLOSE, SEND, SEND\_MAC, SEND\_KEEPchỉ huy). Nếu máy ngang hàng không phản hồi trong thời gian RTR, W5500 sẽ truyền lạigói hoặc các vấn đề hết thời gian chờ.

***RCR (Đăng ký số lần thử lại) [R / W] [0x001B] [0x08]***

RCR định cấu hình số lượng thời gian truyền lại. Khi truyền lại xảy racàng nhiều càng tốt 'RCR + 1', ngắt hết thời gian chờ được đưa ra (Sn\_IR [TIMEOUT] = '1').

***PTIMER (Đăng ký bộ hẹn giờ yêu cầu giao thức điều khiển liên kết PPP) [R / W] [0x001C] [0x0028]***

PTIMER định cấu hình thời gian gửi yêu cầu phản hồi LCP. Đơn vị thời gian là 25ms.

***PMAGIC (Giao thức kiểm soát liên kết PPP Đăng ký số ma thuật) [R / W] [0x001D] [0x00]***

PMAGIC định cấu hình số ma thuật 4byte được sử dụng trong yêu cầu phản hồi LCP.

***PHAR (Đăng ký địa chỉ phần cứng đích ở chế độ PPPoE)[R / W] [0x001E-0x0023] [0x0000]***

PHAR phải được ghi vào địa chỉ phần cứng máy chủ PPPoE có được trong PPPoEquá trình kết nối

***PSID (Đăng ký ID phiên ở chế độ PPPoE) [R / W] [0x0024-0x0025] [0x0000]***

PSID phải được ghi vào ID phiên máy chủ PPPoE có được trong kết nối PPPoEtiến trình.

***PMRU (Đơn vị nhận tối đa ở chế độ PPPoE) [R / W] [0x0026-0x0027] [0xFFFF]***

PMRU định cấu hình đơn vị nhận tối đa của PPPoE.

***UIPR (Đăng ký địa chỉ IP không thể truy cập) [R] [0x0028-0x002B] [0x00000000]***

***UPORTR (Đăng ký cổng không thể kết nối) [R] [0x002C-0x002D] [0x0000]***

W5500 nhận một gói ICMP (Không thể truy cập cổng đích) khi dữ liệu được gửi đếnsố cổng mà ổ cắm không mở và UNREACH bit IR trở thành '1' và UIPR& UPORTR cho biết địa chỉ IP đích và số cổng tương ứng.

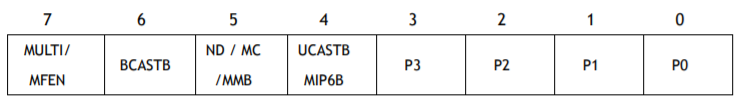
***PHYCFGR (Đăng ký cấu hình W5500 PHY) [R / W] [0x002E] [0b10111XXX]***

PHYCFGR định cấu hình chế độ hoạt động PHY và đặt lại PHY. Ngoài ra, PHYCFGRcho biết trạng thái của PHY như song công, tốc độ, liên kết.

#### **3.4.2. Socket Registers:**

***Sn3\_MR (Thanh ghi chế độ Socket n) [R / W] [0x0000] [0x00]***

Sn\_MR cấu hình tùy chọn hoặc loại giao thức của Socket n.



***Sn\_CR (Thanh ghi lệnh Socket n) [R / W] [0x0001] [0x00]***

Điều này được sử dụng để đặt lệnh cho Socket n như OPEN, CLOSE, CONNECT, LISTEN,Gửi và nhận. Sau khi W5500 chấp nhận lệnh, thanh ghi Sn\_CR làtự động xóa thành 0x00. Mặc dù Sn\_CR được xóa thành 0x00, lệnh vẫnvẫn đang được xử lý. Để kiểm tra xem lệnh đã hoàn thành hay chưa, vui lòngkiểm tra Sn\_IR hoặc Sn\_SR.

***Sn \_IR (Thanh ghi ngắt Socket n) [ RCW1 ] [0x0002] [0x00]***

Sn\_IR cho biết trạng thái của Socket Interrupt chẳng hạn như thiết lập, kết thúc,nhận dữ liệu, thời gian chờ). Khi một ngắt xảy ra và bit tương ứng củaSn\_IMR là '1', bit tương ứng của Sn\_IR trở thành '1'.Để xóa bit Sn\_IR, máy chủ phải ghi bit thành '1'.



***S n \_SR (Thanh ghi trạng thái Socket n) [R] [0x0003] [0x00]***

Sn\_SR cho biết trạng thái của Socket n. Trạng thái của Socket n được thay đổi bởi Sn\_CRhoặc một số gói điều khiển đặc biệt như gói SYN, FIN trong TCP.

***Sn \_PORT (Thanh ghi cổng nguồn Socket n) [R / W] [0x0004-0x0005] [0x0000]***

Sn\_PORT cấu hình số cổng nguồn của Socket n. Nó hợp lệ khi Socket n làđược sử dụng trong chế độ TCP / UDP. Nó phải được đặt trước khi lệnh OPEN được ra lệnh.

***Sn\_DHAR (Thanh ghi địa chỉ phần cứng đích Socket n)[R / W] [0x0006-0x000B] [0xFFFFFFFFFFFF]***

Sn\_DHAR định cấu hình địa chỉ phần cứng đích của Socket n khi sử dụngLệnh SEND\_MAC ở chế độ UDP hoặc nó chỉ ra rằng nó được lấy trong quy trình ARP bởiLệnh CONNECT / SEND.

***Sn\_DIPR (Socket n Đăng ký Địa chỉ IP Đích)[R / W] [0x000C-0x000F] [0x00000000*]**

Sn\_DIPR cấu hình hoặc chỉ ra địa chỉ IP đích của Socket n. Nó hợp lệkhi Socket n được sử dụng trong chế độ TCP / UDP.

Trong chế độ máy khách TCP, nó định cấu hình địa chỉ IP của 'máy chủ TCP' trước khi KẾT NỐIchỉ huy.Trong chế độ máy chủ TCP, nó chỉ ra địa chỉ IP của 'máy khách TCP' sau khi thành côngthiết lập kết nối.Trong chế độ UDP, nó cấu hình một địa chỉ IP của máy ngang hàng để nhận gói UDP bằngLệnh SEND hoặc SEND\_MAC.

***Sn\_DPORT (Thanh ghi cổng đích Socket n) [R / W] [0x0010-0x0011] [0x00]***

Sn\_DPORT cấu hình hoặc chỉ ra số cổng đích của Socket n. Nó hợp lệkhi Socket n được sử dụng trong chế độ TCP / UDP.Trong chế độ máy khách TCP, nó định cấu hình số cổng lắng nghe của 'máy chủ TCP' trước đóLệnh CONNECT.

***Sn\_MSSR (Thanh ghi kích thước phân đoạn tối đa Socket n) [R / W] [0x0012-0x0013] [0x0000]***

Thanh ghi này được sử dụng cho MSS (Kích thước phân đoạn tối đa) của TCP và thanh ghi này hiển thị bộ MSSbởi bên kia khi TCP được kích hoạt ở Chế độ thụ động.

***Sn\_TOS (Loại ổ cắm n IP của đăng ký dịch vụ) [R / W] [0x0015] [0x00]***

Sn\_TOS định cấu hình TOS (trường Loại Dịch vụ trong Tiêu đề IP) của Socket n.Nó được đặt trước lệnh OPEN.

***Sn\_TTL (Thanh ghi TTL Socket n) [R / W] [0x0016] [0x80]***

Sn\_TTL định cấu hình TTL (trường Thời gian tồn tại trong tiêu đề IP) của Socket n.Nó được đặt trước lệnh OPEN.

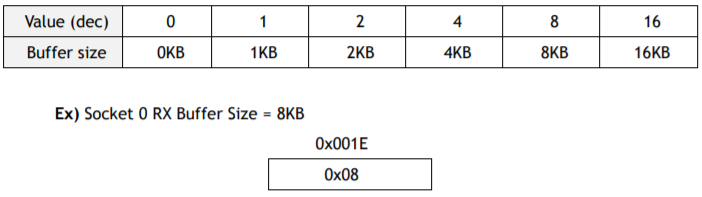
***Sn\_RXBUF\_SIZE (Thanh ghi kích thước bộ đệm Socket n RX) [R / W] [0x001E] [0x02]***

Sn\_RXBUF\_SIZE cấu hình kích thước khối đệm RX của Socket n. Bộ đệm Socket n RXKích thước khối có thể được cấu hình với 1,2,4,8 và 16 Kbyte. Nếu một kích thước khác làđược cấu hình, dữ liệu thường không thể được nhận từ một máy ngang hàng.Mặc dù kích thước Khối đệm Socket n RX ban đầu được định cấu hình thành 2Kbyte, người dùng có thểđịnh cấu hình kích thước của nó bằng Sn\_RXBUF\_SIZE. Tổng tổng của Sn\_RXBUF\_SIZE không đượcvượt quá 16Kbyte. Khi vượt quá, lỗi tiếp nhận dữ liệu sẽ xảy ra.Khi tất cả Sn\_RXBUF\_SIZE đã được định cấu hình, Bộ đệm Socket n RX được cấp phát vớikích thước được định cấu hình trong Bộ nhớ RX 16KB và được gán tuần tự từ Socket 0 đến Socket 7.

***Sn \_TXBUF\_SIZE (Thanh ghi kích thước bộ đệm Socket n TX) [R / W] [0x001F] [0x02]***

Sn\_TXBUF\_SIZE định cấu hình kích thước khối đệm TX của Socket n. Socket n TXKích thước khối đệm có thể được cấu hình với 1,2,4,8 và 16 Kbyte. Nếu một kích thước khácđược định cấu hình, dữ liệu thường không thể được truyền đến một máy ngang hàng.Mặc dù kích thước Khối đệm Socket n TX ban đầu được định cấu hình thành 2Kbyte, người dùng có thểđược định cấu hình lại kích thước của nó bằng Sn\_TXBUF\_SIZE. Tổng số tiền của Sn\_TXBUF\_SIZEkhông được vượt quá 16Kbyte. Khi vượt quá, lỗi truyền dữ liệu làxảy ra.Khi tất cả Sn\_TXBUF\_SIZE đã được định cấu hình, Bộ đệm Socket n TX được cấp phátvới kích thước được định cấu hình trong Bộ nhớ TX 16KB và được gán tuần tự từ Socket 0 đến Socket 7.

Khối đệm Socket n TX có thể được truy cập với phạm vi địa chỉ bù đắp 16bitstừ 0x0000 đến 0xFFFF bất kể kích thước đã định cấu hình. (Tham khảo Sn\_TX\_WR &Sn\_TX\_RD).



***Sn \_TX\_FSR (Đăng ký kích thước miễn phí Socket n TX) [R] [0x0020-0x0021] [0x0800]***

Sn\_TX\_FSR cho biết kích thước trống của Khối đệm Socket n TX. Nó được khởi tạo thànhđược định cấu hình kích thước bởi Sn\_TXBUF\_SIZE. Không nên lưu dữ liệu lớn hơn Sn\_TX\_FSRtrong Bộ đệm Socket n TX vì dữ liệu lớn hơn sẽ ghi đè lên dữ liệu đã lưu trước đóchưa gửi.

***Sn\_TX\_RD (Thanh ghi con trỏ đọc Socket n TX) [R] [0x0022-0x0023] [0x0000]***

Sn\_TX\_RD được khởi tạo bằng lệnh OPEN. Tuy nhiên, nếu Sn\_MR (P [3: 0]) là TCP('0001'), nó được khởi tạo lại trong khi kết nối với TCP.Sau khi khởi tạo, nó sẽ tự động tăng lên bằng lệnh SEND. Gửi lệnhtruyền dữ liệu đã lưu từ Sn\_TX\_RD hiện tại tới Sn\_TX\_WR trong Socket nĐệm TX. Sau khi truyền dữ liệu đã lưu, lệnh SEND tăng

Sn\_TX\_RD giống như Sn\_TX\_WR.Nếu giá trị gia tăng của nó vượt quá giá trị lớn nhất 0xFFFF, (lớn hơn 0x10000và bit mang xuất hiện), sau đó bit mang được bỏ qua và sẽ tự động cập nhậtvới giá trị 16bits thấp hơn.

***Sn \_TX\_WR (Thanh ghi con trỏ ghi Socket n TX) [R / W] [0x0024-0x0025] [0x0000]***

Sn\_TX\_WR được khởi tạo bằng lệnh OPEN. Tuy nhiên, nếu Sn\_MR (P [3: 0]) là TCP('0001'), nó được khởi tạo lại trong khi kết nối với TCP.

Nó nên được đọc hoặc được cập nhật như sau.

1. Đọc địa chỉ bắt đầu để lưu dữ liệu truyền.

2. Lưu dữ liệu truyền từ địa chỉ bắt đầu của Socket n TX đệm.

3. Sau khi lưu dữ liệu truyền, hãy cập nhật Sn\_TX\_WR vàogiá trị tăng nhiều như kích thước dữ liệu đang truyền. Nếu giá trị giatăngvượt quá giá trị tối đa 0xFFFF (lớn hơn 0x10000 và mangbit xảy ra), sau đó bit mang được bỏ qua và sẽ tự động cập nhậtvới giá trị 16bits thấp hơn.

4. Truyền dữ liệu đã lưu trong Socket n TX Buffer bằng cách sử dụng SEND/ SEND chỉ huy.

***Sn\_RX\_RSR (Thanh ghi kích thước đã nhận Socket n) [R] [0x0026-0x0027] [0x0000]***

Sn\_RX\_RSR cho biết kích thước dữ liệu nhận được và lưu trong Socket n RX Buffer. Sn\_RX\_RSRkhông vượt quá Sn\_RXBUF\_SIZE và được tính là chênh lệch giữa'Con trỏ ghi Socket n RX (Sn\_RX\_WR)' và 'Con trỏ đọc Socket n RX (Sn\_RX\_RD)'.

***S n \_RX\_RD (Thanh ghi con trỏ dữ liệu đọc Socket n RX) [R / W] [0x0028-0x0029] [0x0000]***

Sn\_RX\_RD được khởi tạo bằng lệnh OPEN. Đảm bảo được đọc hoặc cập nhật dưới dạng

theo sau.

1. Đọc địa chỉ lưu bắt đầu của dữ liệu đã nhận

2. Đọc dữ liệu từ địa chỉ bắt đầu của Socket n RX Buffer.

3. Sau khi đọc dữ liệu đã nhận, Cập nhật Sn\_RX\_RD thành giá trị tăng lên càng nhiềunhư kích thước đọc. Nếu giá trị gia tăng vượt quá giá trị lớn nhất 0xFFFF,nghĩa là lớn hơn 0x10000 và xảy ra bit mang, hãy cập nhật với giá trị thấp hơnGiá trị 16bits bỏ qua bit mang.

4. Lệnh RECV để thông báo Sn\_RX\_RD cập nhật cho W5500.

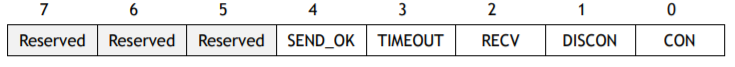
***Sn\_RX\_WR (Thanh ghi con trỏ ghi Socket n RX) [R] [0x002A-0x002B] [0x0000]***

Sn\_RX\_WR được khởi tạo bằng lệnh OPEN và nó được tự động tăng lên theo dữ liệuthu nhận.Nếu giá trị tăng lên vượt quá giátrị lớn nhất 0xFFFF, (lớn hơn 0x10000và bit mang xuất hiện), sau đó bit mang được bỏ qua và sẽ tự động cập nhậtvới giá trị 16bits thấp hơn.

***Sn\_IMR (Thanh ghi mặt nạ ngắt Socket n) [R / W] [0x002C] [0xFF]***

Sn\_IMR che dấu ngắt của Socket n. Mỗi bit tương ứng với mỗi bit của Sn\_IR.Khi một Ngắt Socket n xảy ra và bit tương ứng của Sn\_IMR là '1',bit tương ứng của Sn\_IR trở thành '1'.Khi cả bit tương ứng của Sn\_IMR và Sn\_IR là '1' và bit thứ n của SIR

là '1', Máy chủ lưu trữ bị gián đoạn bởi mã PIN INTn được xác nhận xuống mức thấp.



***Sn\_FRAG (Thanh ghi Socket n Fragment) [R / W] [0x002D-0x002E] [0x4000]***

Sn\_FRAG định cấu hình FRAG (Trường mảnh trong tiêu đề IP).

***Sn\_KPALVTR (Socket n Keep Alive Time Register) [R / W] [0x002F] [0x00]***

Sn\_KPALVTR định cấu hình bộ đếm thời gian truyền của gói 'KEEP ALIVE (KA)' củaỔ khóa. Nó chỉ hợp lệ trong chế độ TCP và bị bỏ qua trong các chế độ khác. Đơn vị thời gian là5 giây.Gói KA có thể truyền sau khi Sn\_SR được thay đổi thành SOCK\_ESTABLISHED và sau đódữ liệu được truyền hoặc nhận đến / từ một máy ngang hàng ít nhất một lần.Trong trường hợp 'Sn\_KPALVTR> 0', W5500 tự động truyền gói KA sau thời gian-khoảng thời gian để kiểm tra kết nối TCP (Quá trình tự động giữ lại).Trong trường hợp 'Sn\_KPALVTR = 0', quá trình tự động giữ lại sự sống sẽ không hoạt động và gói KAcó thể được máy chủ truyền bằng lệnh SEND\_KEEP (Quy trình thủ công-giữ-tồn-tại).Quy trình thủ công-giữ-tồn-tại bị bỏ qua trong trường hợp 'Sn\_KPALVTR> 0'.

# **III. MODULE ETHERNET W5500–GIAO THỨC TCP/IP Stack51/STM32/SPI:**

## **1. Tìm hiểu chung về các giao thức:**

### **1.1. TCP/IP:**

**Tổng quan:**

- TCP/ IP (Transmission Control Protocol/ Internet Protocol - Giao thức điều khiển truyền nhận/ Giao thức liên mạng), là một bộ giao thức trao đổi thông tin được sử dụng để truyền tải và kết nối các thiết bị trong mạng Internet. TCP/IP được phát triển để mạng được tin cậy hơn cùng với khả năng phục hồi tự động.

- Ý tưởng hình thành mô hình TCP/IP được bắt nguồn từ Bộ giao thức liên mạng trong công trình DARPA vào năm 1970. Trải qua vô số năm nghiên cứu và phát triển của 2 kỹ sư Robert E. Kahn và Vinton Cerf cùng sự hỗ trợ của không ít các nhóm nghiên cứu. Đầu năm 1978, giao thức TCP/ IP được ổn định hóa với giao thức tiêu chuẩn được dùng hiện nay của Internet đó là mô hình TCP/IP Version 4.

- Vào năm 1975, cuộc thử nghiệm thông nối giữa 2 mô hình TCP/IP được diễn ra thành công. Cũng bắt đầu từ đây, cuộc thử nghiệm thông nối giữa các mô hình TCP/IP được diễn ra nhiều hơn và đều đạt được kết quả tốt, từ đây giao thức và mô hình TCP/IP được phổ biến rộng rãi trên khắp thế giới.

- Các giao thức phổ biến của TCP/IP gồm:

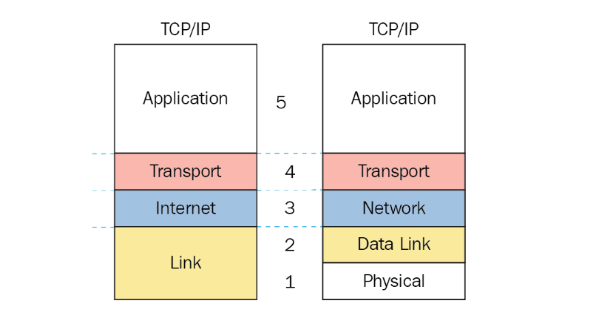
+ HTTP (Hyper Text Transfer Protocol): Xử lý giao tiếp giữa máy chủ web và trình duyệt web.

+ HTTPS (Secure HTTP): Xử lý giao tiếp bảo mật giữa máy chủ web và trình duyệt web.

+ FTP (File Transfer Protocol – Giao thức truyền tệp): Xử lý việc truyền tải file giữa các máy tính.

**Cách thức và mô hình hoạt động:**

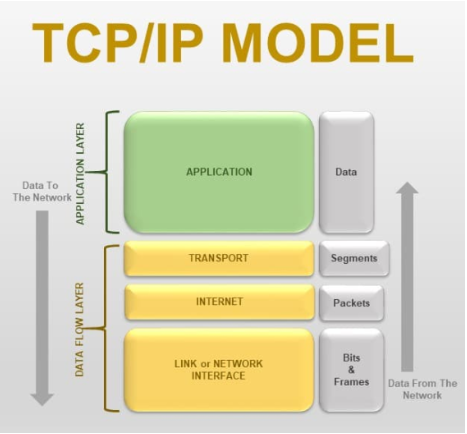
- Phân tích từ tên gọi, TCP/IP là sự kết hợp giữa 2 giao thức. Trong đó IP (Giao thức liên mạng) cho phép các gói tin được gửi đến đích đã định sẵn, bằng cách thêm các thông tin dẫn đường vào các gói tin để các gói tin được đến đúng đích đã định sẵn ban đầu. Và giao thức TCP (Giao thức truyền vận) đóng vai trò kiểm tra và đảm bảo sự an toàn cho mỗi gói tin khi đi qua mỗi trạm. Trong quá trình này, nếu giao thức TCP nhận thấy gói tin bị lỗi, một tín hiệu sẽ được truyền đi và yêu cầu hệ thống gửi lại một gói tin khác. Quá trình hoạt động này sẽ được làm rõ hơn ở chức năng của mỗi tầng trong mô hình TCP/IP.

- Một mô hình TCP/IP tiêu chuẩn bao gồm 4 lớp được chồng lên nhau, bắt đầu từ tầng thấp nhất là **Tầng vật lý (Physical)** → **Tầng mạng (Network)** → **Tầng giao vận (Transport)** và cuối cùng là **Tầng ứng dụng (Application)**

**Tầng 4 – Tầng ứng dụng (Application)**

- Đây là lớp giao tiếp trên cùng của mô hình. Đúng với tên gọi, tầng Ứng dụng đảm nhận vai trò giao tiếp dữ liệu giữa 2 máy khác nhau thông qua các dịch vụ mạng khác nhau (duyệt web, chat, gửi email, một số giao thức trao đổi dữ liệu: SMTP, SSH, FTP,...).

- Dữ liệu khi đến đây sẽ được định dạng theo kiểu Byte nối Byte, cùng với đó là các thông tin định tuyến giúp xác định đường đi đúng của một gói tin.



**Tầng 3 – Tầng giao vận (Transport)**

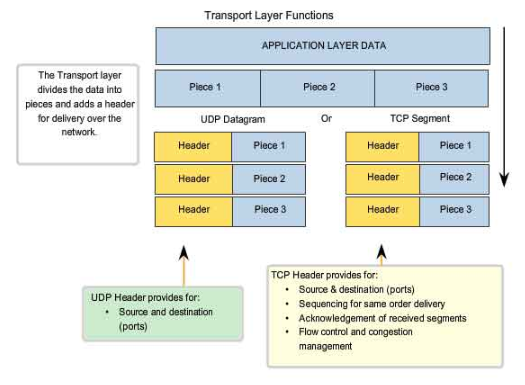
- Chức năng chính của tầng 3 là xử lý vấn đề giao tiếp giữa các máy chủ trong cùng một mạng hoặc khác mạng được kết nối với nhau thông qua bộ định tuyến.

- Tại đây dữ liệu sẽ được phân đoạn, mỗi đoạn sẽ không bằng nhau nhưng kích thước phải nhỏ hơn 64KB.

- Cấu trúc đầy đủ của một Segment lúc này là Header chứa thông tin điều khiển và sau đó là dữ liệu.

- Trong tầng này còn bao gồm 2 giao thức cốt lõi là TCP và UDP. Trong đó, TCP đảm bảo chất lượng gói tin nhưng tiêu tốn thời gian khá lâu để kiểm tra đầy đủ thông tin từ thứ tự dữ liệu cho đến việc kiểm soát vấn đề tắc nghẽn lưu lượng dữ liệu.

- Trái với điều đó, UDP cho thấy tốc độ truyền tải nhanh hơn nhưng lại không đảm bảo được chất lượng dữ liệu được gửi đi.

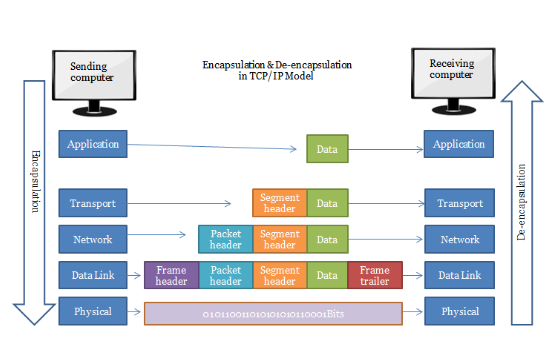


**Tầng 2 – Tầng mạng (Internet)**

- Tại đây, nó cũng được định nghĩa là một giao thức chịu trách nhiệm truyền tải dữ liệu một cách logic trong mạng.

- Các phân đoạn dữ liệu sẽ được đóng gói (Packets) với kích thước mỗi gói phù hợp với mạng chuyển mạch mà nó dùng để truyền dữ liệu.

- Lúc này, các gói tin được chèn thêm phần Header chứa thông tin của tầng mạng và tiếp tục được chuyển đến tầng tiếp theo. Các giao thức chính trong tầng là IP, ICMP và ARP.



**Tầng 1 – Tầng vật lý (Physical):**

- Là sự kết hợp giữa tầng Vật lý và tầng liên kết dữ liệu của mô hình OSI. Chịu trách nhiệm truyền dữ liệu giữa hai thiết bị trong cùng một mạng. Tại đây, các gói dữ liệu được đóng vào khung (gọi là Frame) và được định tuyến đi đến đích đã được chỉ định ban đầu.

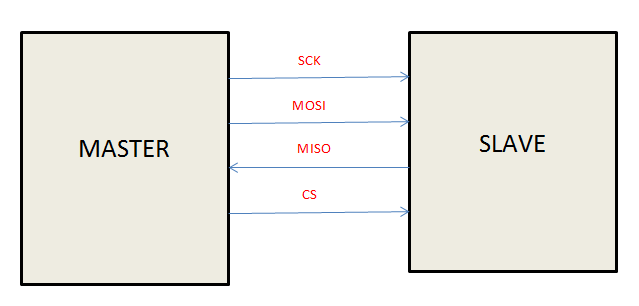
### **1.2. STM32/SPI:**

**Tổng quan về giao tiếp SPI:**

- SPI là một giao thức giao tiếp phổ biến được sử dụng bởi nhiều thiết bị khác nhau.

- SPI có thể truyền dữ liệu mà không bị gián đoạn. Bất kỳ số lượng bit nào cũng có thể được gửi hoặc nhận trong một luồng liên tục. Với I2C và UART, dữ liệu được gửi dưới dạng gói, giới hạn ở một số bit cụ thể. Điều kiện bắt đầu và dừng xác định điểm bắt đầu và kết thúc của mỗi gói, do đó dữ liệu bị gián đoạn trong quá trình truyền.

- Các thiết bị giao tiếp qua SPI có quan hệ master - slave. Master là thiết bị điều khiển (thường là vi điều khiển), còn slave (thường là cảm biến, màn hình hoặc chip nhớ) nhận lệnh từ master. Cấu hình đơn giản nhất của SPI là hệ thống một slave, một master duy nhất, nhưng một master có thể điều khiển nhiều hơn một slave.



***MOSI*** *(đầu ra master / đầu vào slave) - đường truyền cho master gửi dữ liệu đến slave.*

***MISO*** *(đầu vào master / đầu ra slave) - đường cho slave gửi dữ liệu đến master.*

***SCLK*** *(clock) - đường cho tín hiệu xung nhịp.*

***SS / CS*** *(Slave Select / Chip Select) - đường cho master chọn slave nào để gởi tín hiệu.*

**Cách hoạt động:**

- Tín hiệu xung nhịp đồng bộ hóa đầu ra của các bit dữ liệu từ master để lấy mẫu các bit của slave. Một bit dữ liệu được truyền trong mỗi chu kỳ xung nhịp, do đó tốc độ truyền dữ liệu được xác định bởi tần số của tín hiệu xung nhịp. Giao tiếp SPI được khởi tạo bởi master kể từ khi master cấu hình và tạo ra tín hiệu xung nhịp.

- Tín hiệu xung nhịp trong SPI có thể được sửa bằng cách sử dụng các thuộc tính của phân cực xung nhịp và pha xung nhịp. Hai thuộc tính này làm việc cùng nhau để xác định khi nào các bit được xuất ra và khi được lấy mẫu. Phân cực xung nhịp có thể được thiết lập bởi master để cho phép các bit được xuất ra và lấy mẫu trên cạnh lên hoặc xuống của chu kỳ xung nhịp. Pha xung nhịp có thể được đặt để đầu ra và lấy mẫu xảy ra trên cạnh đầu tiên hoặc cạnh thứ hai của chu kỳ xung nhịp, bất kể nó đang tăng hay giảm.

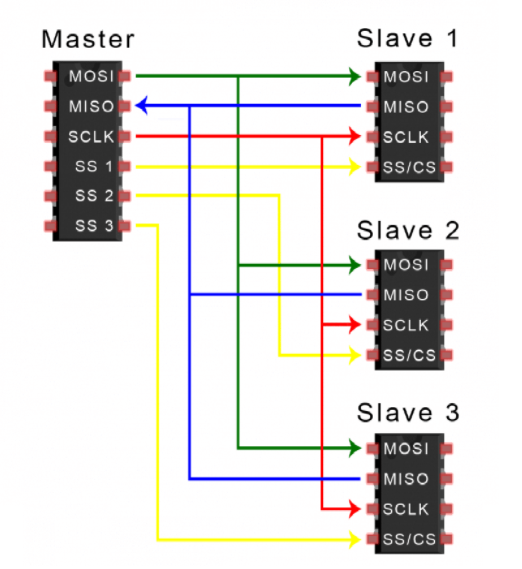
- Master có thể chọn slave mà nó muốn giao tiếp bằng cách đặt đường CS / SS của slave ở mức điện áp thấp. Ở trạng thái idle, không truyền tải, dòng slave select được giữ ở mức điện áp cao. Nhiều chân CS / SS có thể có sẵn trên thiết bị master cho phép đấu dây song song nhiều slave. Nếu chỉ có một chân CS / SS, nhiều slave có thể được kết nối với master bằng cách nối chuỗi.

- Master gửi dữ liệu đến slave từng bit, nối tiếp qua đường MOSI. Slave nhận dữ liệu được gửi từ master tại chân MOSI. Dữ liệu được gửi từ master đến slave thường được gửi với bit quan trọng nhất trước.

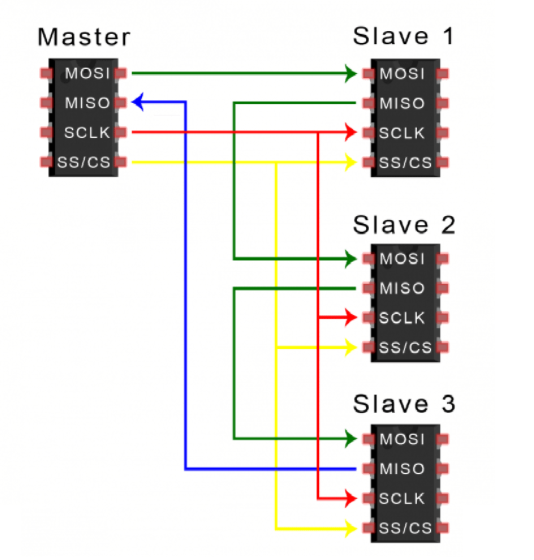
- Slave cũng có thể gửi dữ liệu trở lại master thông qua đường MISO nối tiếp. Dữ liệu được gửi từ slave trở lại master thường được gửi với bit ít quan trọng nhất trước.

- Khi nhiều slave có hai cách để kết nối:

+ Nếu chỉ có một chân CS / SS, nhiều slave có thể được kết nối với master bằng cách nối chuỗi.



- Nếu chỉ có một chân slave select, các slave có thể được nối chuỗi như sau:



- Các bước truyền nhận dữ liệu:

Master ra tín hiệu xung nhịp.

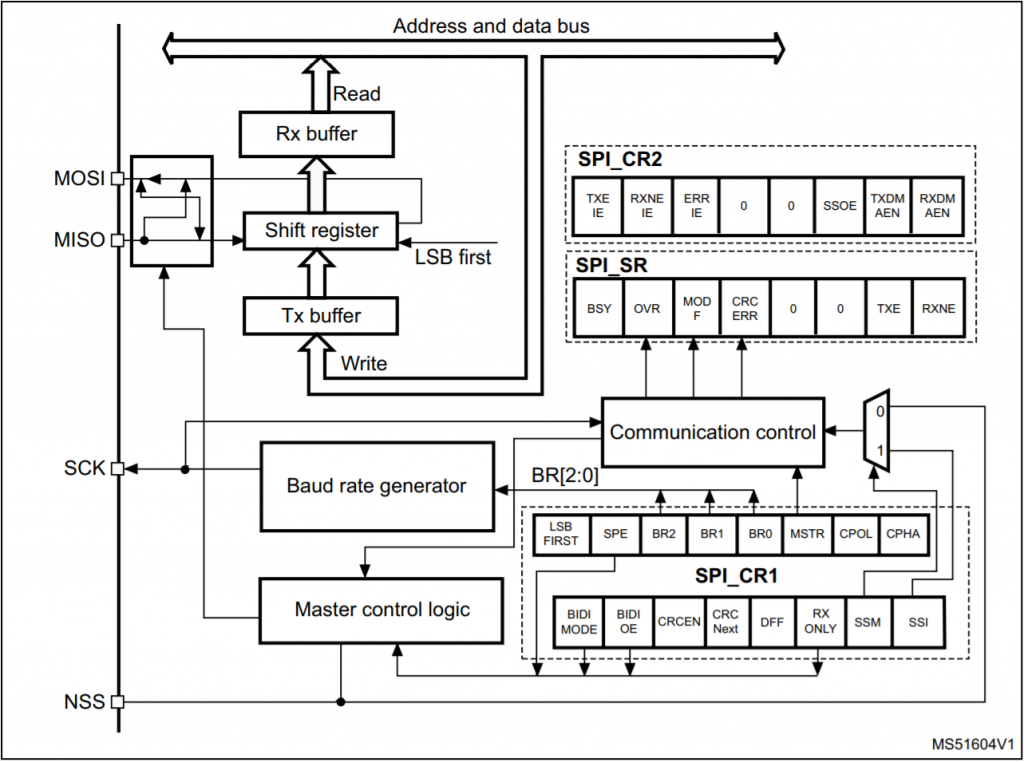
→Master chuyển chân SS / CS sang trạng thái điện áp thấp, điều này sẽ kích hoạt slave.

→ Master gửi dữ liệu từng bit một tới slave dọc theo đường MOSI. Slave đọc các bit khi nó nhận được.

→ Nếu cần phản hồi, slave sẽ trả lại dữ liệu từng bit một cho master dọc theo đường MISO. Master đọc các bit khi nó nhận được.

**SPI trong STM32:**

- Sơ đồ khối:



- STM32F4 cung cấp 5 khối SPI (SPI1, SPI2, SPI3, SPI4, SPI5) để người dùng kết nối với các ngoại vi giao tiếp bằng SPI.

- Theo như lý thuyết, khối SPI trên STM32F4 gồm:

+ 4 chân để kết nối với các thiết bị bên ngoài MOSI, MISO, SCK và NSS.

+ Thanh ghi dịch – Shift Register để thực hiện phép dịch trong quá trình truyền và nhận dữ liệu giữa Master và Slave.

+ Các thanh ghi SPI\_CR1, SPI\_CR2 và SPI\_SR cho phép cấu hình và trạng thái.

+ Có thể làm việc với SPI qua 3 chế độ: Polling, Interrupt và DMA.

## **2. Công nghệ & Tính năng sản phẩm thực tế:**

Như đã tìm hiểu rất kĩ về STM32 và đặc biệt là W5500 ở phần trên, tới đây chúng ta sẽ cùng tìm hiểu về sản phẩm được ứng dụng công nghệ trên vào trong thực tế đời sống đó là Module Ethernet W5500 sử dụng giao thức TCP/IP Stack51/STM32/SPI.



*Module Ethernet W5500 SPI*

* Module Ethernet W5500 sử dụng giao thức (TCP/IP Stack51/STM32/SPI) là một mạch giao tiếp sử dụng SPI để kết nối vi điều khiển với Ethernet hiệu quả, ổn định.
* Module Ethernet W5500 sử dụng chip mới W5500 (phân tích ở phần 2) với hai kiểu giao tiếp TCP / IP cho phép kết nối Internet dễ dàng hơn trong hệ thống nhúng sử dụng SPI (giao diện ngoại vi nối tiếp).
* Module hỗ trợ tốc độ 80GHz và giao thức SPI mang lại hiệu quả cao, do đó người dùng có thể giao tiếp với mạng tốc độ cao.
* W5500 cung cấp chức năng WOL (Wake On Lan) và chế độ tắt nguồn, giúp tiết kiệm năng lượng.

## **3. Thông số kỹ thuật sản phẩm:**

Thông số kỹ thuật chính của Module Ethernet W5500:

* IC chính: IC W5500 từ Wiznet sản xuất.
* Điện áp nguồn: 5V hoặc 3.3VDC; Điện áp đầu ra 3.3V
* Dòng cho phép: 128 mA
* Mức tín hiệu giao tiếp SPI TTL
* Hỗ trợ các giao thức: TCP, UDP, ICMP, IPv4, ARP, IGMP, PPPoE
* Hỗ trợ 8 ổ cứng độc lập sử dụng đồng thời
* Hỗ trợ chế độ tiết kiệm điện
* Hỗ trợ Wake với cổng LAN qua UDP
* Hỗ trợ giao diện ngoại vi nối tiếp tốc độ cao (SPI MODE 3,0)
* Bộ nhớ trong 32Kbyte cho bộ đệm Rx / Tx.
* Hỗ trợ giao tiếp tự động song cổng và bán song cổng
* Không hỗ trợ phân mảnh IP
* Đầu ra LED quang (Full/Half duplex, Link, Speed, Active)
* 48 chân LQHP không chứa chì độc hại (7x7mm, 0.5mm)
* Tần số xung clock tối đa: 80MHz
* Kích thước& trọng lượng Module: 55mm x 28mm; 15g

# **IV. TÀI LIỆU THAM KHẢO:**

* Tóm tắt đặc điểm kĩ thuật: [Cortex-M3 Specification Summary; ARM Holdings](http://arm.com/products/processors/cortex-m/cortex-m3.php?tab=Specifications)
* Tóm tắt đặc điểm kĩ thuật: [Cortex-M7 Specification Summary; ARM Holdings](http://arm.com/products/processors/cortex-m/cortex-m7-processor.php?tab=Specifications)
* [Trang web của Ban khám phá STM32; STMicroelectronics](https://www.st.com/en/evaluation-tools/stm32-mcu-discovery-kits.html)
* W5500-Datasheet: https://cdn.sparkfun.com/datasheets/Dev/Arduino/Shields/W5500\_datasheet\_v1.0.2\_1.pdf
* Code mẫu Arduino: <https://github.com/Wiznet/WIZ_Ethernet_Library>
* *The Insider's Guide To The STM32 ARM Based Microcontroller*; 2nd Edition (v1.8); Trevor Martin; Hitex; 96 pages; 2009
* *µC/TCP-IP: The Embedded Protocol Stack for the STMicroelectronics STM32F107*; 1st Edition; Christian Légaré; Micrium; 824 pages; 2010